Hardware Implementation of an Efficient Key Management Scheme for Wireless Sensor Networks

Jilna P., Deepthi P.P
Department of ECE
National Institute of Technology
Calicut, India

Jayaraj U.K.
National Institute of Electronics and Information Technology
Calicut, India

Abstract

This paper presents the design and implementation of an efficient elliptic curve (EC) based key management scheme for wireless sensor networks in applications with high security concerns. The schemes available in literature are well analyzed and integrated architecture for two EC based key management protocols are developed and compared. The basic functions of authentication and random number generation are implemented using EC based algorithms instead of standard algorithms such as SHA or AES for optimized design of overall structure through hardware reuse. The proposed architectures are implemented over GF(2^{163}) on a Kintex 7 FPGA board using Xilinx ISE. The EC point multiplication is done using Lopez- Dahab algorithm and the finite field elements are represented using Gaussian normal basis to reduce the computational complexity and resist side channel attack.

1. Introduction

Wireless sensor networks consists of a set of sensor nodes, each equipped with one or more sensors, limited communication capabilities, storage and processing resources. These networks are deployed in many real-world applications such as health care, defense applications, habitat monitoring etc. [1], [2]. Since security was not a major concern when the WSN was developed, the sensor nodes do not have any built-in security protocols. Due to the resource constrained nature, the WSNs make use of symmetric key algorithms to secure the data. Various key establishment schemes intended for wireless sensor networks are available in literature. These schemes are evaluated in terms of computational cost, communication overhead, storage efficiency, resilience and scalability. For applications with strong security concerns, ECC is a promising solution and a number of key establishment and authentication proposals based on ECC are present in literature [3],[4]. All key establishment schemes involve the computation of a message authentication code (MAC) for mutual authentication. The MAC is usually implemented using standard algorithms such as SHA or AES. These cryptographic algorithms can be implemented either in software or hardware. In [5] the authors have shown that software implementation of elliptic curve (EC) based cryptographic algorithms are not efficient as they consume large energy, affect the execution of other programs running on the node and are also prone to attacks such as cache collision timing attacks. They have proposed a custom hardware platform which includes hardware units for ECC, AES and SHA. Various optimized architectures for each of these algorithms are available in literature [6], [7]. But implementing key exchange and MAC using two different cryptographic primitives increases the overall structural complexity of the system.

In this paper, hardware architecture for two key establishment schemes based on elliptic curves is implemented and compared. EC based algorithms which we have designed for authentication [8] and random number generation [9] in resource constrained applications are used for implementation instead of standard algorithms to reduce the structural complexity through hardware reuse.

The remainder of this paper is organized as follows. Section 2 briefly describes the theoretical background. The proposed architectures are presented in Section 3. In Section 4, the flow chart is presented and in Section 5, the details of hardware architecture and implementation results are given. In Section 6, some conclusions are drawn.

2. Theoretical background

The use of symmetric key algorithms for data encryption and authentication demand efficient key establishment protocols which are suitable for the resource constrained nature of WSNs. Other than the conventional schemes, many light weight protocols targeting the WSN are available in the literature. These protocols are evaluated based on the efficiency, security and flexibility metric. The key exchange protocols can be broadly classified as key pre-deployment scheme or key derivation information pre-deployment scheme. Light weight protocols such as LEAP (Localised encryption and authentication protocol), random key pre-distribution scheme, polynomial based key distribution etc.

[10],[11] offer good performance when evaluated in terms of efficiency metric but provide no security if a node is compromised in the initialization phase.

The security analysis of various key establishment techniques shows that schemes dependent on hard mathematical problems are more secure than light weight techniques. ECC based key management techniques and modular arithmetic based key management (MAKM) [12] offer high security even if the node is compromised in the initialization phase. ECC is often preferred because of the increased security per bit of the key.

3. Proposed hardware architecture for key establishment in WSNs

In this section, key management schemes and the algorithms used for implementing the authentication code and random number generation required for key establishment are presented.

3.1. Key exchange protocol

In [3], the authors have presented two EC based key management schemes for distributed sensor networks. One is a hybrid of symmetric and asymmetric cryptography (EC-H) and the other is a fully asymmetric protocol (EC-A). Both methods offer high resilience in comparison with other light weight schemes even if nodes are compromised in the initialization phase. But the storage requirement prior to deployment is high and limits the number of node addition phases that it can support. Hence a modified EC-A algorithm is used for implementation such that the storage requirement is reduced to support any number of node addition phases.

Algorithm 1. Modified EC-A algorithm for key establishment

Input: P = (x, y) in E(GF(2^n)), node id ID_A, network wide symmetric key K, secret key of certification authority q_{CA}

Output: Shared secret key K_{AB}.

1. Generate a random number g_{Ai} in GF(2^{163}) and compute $IC_{Ai} = \{g_{Ai}P,i,ID_A,t_x\}$ where i is the generation and t_x is the expiration time.
2. Compute $q_{Ai} = g_{Ai} + e_{Ai}q_{CA}$ where $e_{Ai} = MAC(IC_{Ai})$. and store in memory.
3. Broadcast a random nonce N_A, IC_{Ai}, $MAC(N_A,IC_{Ai})$ and wait for acknowledgement.
4. Verify the acknowledgement (MAC).
5. Read $g_{Bi}P$ from the acknowledgement received and compute $Q_{Bi} = g_{Bi}P + e_{Bi}q_{CA}$ P.
6. Compute $K_{AB} = kdf(q_{Ai}Q_{Bi}, N_A, N_B)$.
7. Use MAC computed on the derived key for the confirmation.
8. If confirmation succeeds then both node accepts the key as pair wise key and the N_A, N_B values are deleted immediately.

Considering an implementation of EC-A over GF(2^{163}), the values stored in a sensor node prior to deployment are, point P (40 bytes), Network wide Symmetric key K (20 bytes), Implicit certificate of the node IC_x (44 bytes), Public key of Certification Authority (40 bytes) and Secret EC key of the node q_x (20 bytes). In addition, each node should store EC key (20 bytes) and Implicit Certificate (44 bytes) for each node addition phase. Thus the total storage requirement is 164 + 64 N bytes where N is the number of node addition phases. In the modified algorithm, the values stored are point P (40 bytes), network wide symmetric key K (20 bytes) and secret key of certification authority q_{CA} (20 bytes). The system makes use of a pseudo random sequence generator (PRSG) to generate the random numbers. An EC based PRSG is chosen so that it reuses the resources already available and doesn't add much to the structural complexity of the system.

Even though the storage efficiency of modified EC-A is more compared to EC-A, the communication overhead remains the same (192 bytes considering an implementation over GF(2^{163})). Assuming a cost of 185nJ/bit for transmitting and 133nJ/bit for receiving [13], the modified EC-A requires approximately 31mJ for a pair wise secret key establishment and hence not energy efficient. In [14] we have proposed an efficient elliptic curve based key establishment technique for WSNs (hereinafter called ECKM) with reduced communication overhead and storage requirement. A detailed analysis and comparison of the ECKM with other key exchange methods available in the literature is done in terms of the security, efficiency and flexibility metric. The communication overhead, computational cost and the number of transmissions required for this method is much less compared to the EC based methods available in the literature. Similarly in comparison with other light weight schemes (Random key pre-distribution, LEAP etc.) the resilience provided by ECKM is very high. Hence this method of key establishment is used for implementation and compared with implementation of Algorithm 1.

Algorithm 2: ECKM

Input: P = (x, y) in E(GF(2^n)), node id ID_A, random integer n_A, master key MK.

Output: Shared secret key K_{AB}.

1. Compute n_AP.
2. Compute $S_A = MAC_{MK}(n_AP)$.

3. Broadcast ID_A and n_AP and wait for acknowledgement.
4. Read n_BP from the acknowledgement received and compute n_An_BP.
5. Compute shared secret $K_{AB} = MAC_{MK}(n_An_BP)$.
6. Return K_{AB}

In ECKM, the stored parameters are the node id, the point P on the elliptic curve, the random integer, master key and the time out period which are loaded into the node memory prior to deployment. Considering an implementation over $GF(2^{163})$ the communication overhead of ECKM is only 100 bytes (approximately half of modified EC-A). The number of transmissions and computational complexity of ECKM is also much less when compared to modified EC-A protocol. Both schemes have similar storage requirement and security.

3.2. Message authentication code

The message authentication code, which is a part of every key establishment technique, can be implemented using a block cipher algorithm (AES) or keyed hash functions (HMAC). But the implementation of these standard algorithms increases the hardware requirement. To reduce the hardware resource requirement, an EC based MAC presented in [8] is used for implementing the integrated system. In [8] we have implemented a prototype of EC based MAC and shown that the hardware required for implementation is very less compared to SHA or AES if the EC point multiplication unit is already available as a part of key exchange. Hence the algorithm presented in [8] is used for implementing the MAC function in both the key establishment schemes in such a way that it time shares the point multiplication unit used for key exchange.

Message authentication code generation consists of two stages. In the first stage, the message to be authenticated M, is compressed using modular division to generate a residue. The residue is given as input to the one way function of point multiplication to generate the authentication tag. The detailed algorithm for MAC generation is given below [8]. For implementation, the two generator polynomials are pre-loaded into the memory and the master key MK is taken as the secret integer k

Algorithm 3: Algorithm for MAC generation

Input: P = (x, y) in $E(GF(2^n))$, a secret integer k, message polynomial M(x).

Output: MAC(M).

1. Perform modular division of M(x) using two generator polynomials $g_1(x)$ and $g_2(x)$ of degree n which are kept secret to obtain residues r_1 and r_2 in n bits.
2. Compute $r = r_1 + r_2$
3. Compute R = rkP.
4. h = tr (X(R))
5. Return h

3.3. Pseudo random number generator

In the modified-EC-A algorithm, multiple EC key pairs are replaced with a random number generator to reduce the storage requirement and to support any number of node addition phases. In addition the key establishment process involves exchange of random nonce for mutual authentication. The authors suggest AES for use as random number generator. In the proposed hardware architecture an EC based random number generator in [9] is used so that the overall hardware complexity of the key establishment system can be reduced by time sharing the point multiplication unit.

Algorithm 4: Algorithm for pseudo random sequence generation

Input: $P = (x,y) \in E(GF(2^n))$, secret key 'e' of length 2n bits
Output: Pseudo random bit sequence s_i

1. Get e_1 and e_2 from e by truncating it to required no. of bits.
2. $k_0 = e_1$ and seed of LFSR $C_0 = e_2$.
3. Key for ith iteration $k_i = X(k_{i-1}P) + C_{i-1}$.
4. Advance the LFSR count to next state C_i.
5. Compute the ith output point $S_i = k_iP + e_1P$.
6. Truncate $X(S_i)$ to generate output bit sequence $s_i = trunc(X(S_i))$.
7. return s
8. Go back to step 3

4. Flow chart of the proposed systems

In the proposed architectures, a network of n sensor nodes is considered where each node is initialized with different time delay. The sequence of operations for the two key establishment algorithms is explained below.

4.1. Flow chart of modified EC-A algorithm

In the modified EC-A algorithm, when a node is initialized, it generates a random number g_{Ai} and computes the implicit certificate (IC) as $IC_{Ai} = \{g_{Ai}P,i,ID_A,t_x\}$ where i is the generation and t_x is the expiration time. The private key of the node corresponding to this implicit certificate is computed as $q_{Ai} = g_{Ai} + e_{Ai}\, q_{CA}$, where $e_{Ai} = MAC(IC_{Ai})$ and q_{CA} is the private key of the certification authority. The node then broadcasts its IC, random nonce N_A

and a MAC for verification. On receiving the ACK, the node verifies the MAC and if successful, computes the shared secret key. The node sends a new ACK generated using the shared secret key for verification by the neighbouring node and similarly verifies the ACK send by the neighbour. The node now checks for ACK signals from other neighbours. The sequence of operations is shown in Figure 1.

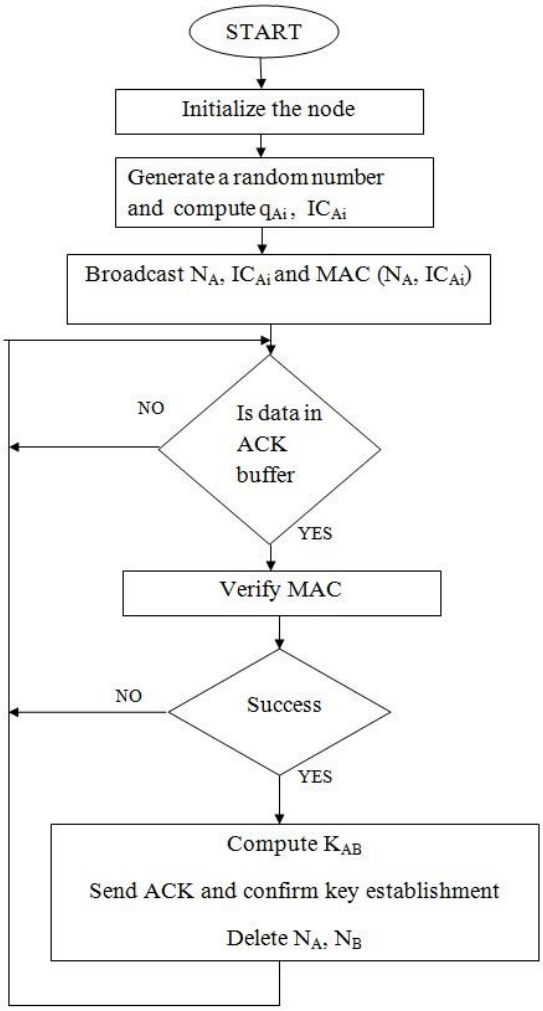

Figure 1. Flow chart of modified EC-A algorithm

4.2. Flow chart of ECKM algorithm

Prior to deployment, the system memory is loaded with a random integer n_A, a point on the elliptic curve P, the master key MK, node id ID_A and two polynomials for authentication. When initialized, the system reads the n_A and P values from the memory and computes the point n_AP. Once the computation is complete, the node broadcasts its ID and the point n_AP which is now stored in memory. The node now waits for the acknowledgement (ACK) signal from the neighbouring nodes. The n nodes in the network are initialized with different time delays so that the ACK signal is received at different time intervals. Simultaneously the phase I MAC unit is enabled to initiate the computation of the secret value S_A; the value S_A is stored in the memory (to facilitate key establishment in the working phase) and the phase I MAC unit is disabled. In the meantime, the ACK signals received are stored in a buffer. The size of the buffer is determined by the number of neighbouring nodes in the network. From the ACK signal received, the n_BP value is read and the point n_An_BP is computed. The shared secret key is generated using the MAC function and the received MAC is verified. After verification, the shared secret key and corresponding node id are stored in the memory. This process is continued until all the ACK signals in the buffer are processed.

5. Integrated hardware architecture

The hardware architecture of the proposed integrated systems is given in Fig 3. The various units are (i) Elliptic curve point multiplication (ECPM) unit (ii) Phase I of MAC unit (iii) Acknowledgement (ACK) buffer (iv) Broadcast unit (v) Memory and (vi) System controller. The hardware for modified EC-A requires an additional unit for generating the pseudo random sequence. The implementation is done over $GF(2^{163})$ for 80 bit security as recommended by NIST. The standard elliptic curve $E : y^2 + xy = x^3 + 1$ is used for implementation. The pre-deployment values for both the schemes are stored in the memory. The ECPM unit, ACK buffer, Phase I of MAC, pseudo random sequence generation unit and the broadcast unit are connected to the memory through data bus. As the implementation is done over $GF(2^{163})$, the size of the data bus is customized as 163 bits. The address bus is connected between the controller and memory. The controller places the address in the address bus to read/write the data from/to memory and generates the control signals such that the specific unit can read/write data from/to the data bus. The size of the address bus is customized as 8 bits. The detailed architecture of various hardware units are explained below.

5.1. ECPM unit

Double and add and Lopez-Dahab are the two popular algorithms for point multiplication available in the literature. In the double and add algorithm, the number of computations required in each iteration depends on the key bit and is easily prone to power analysis attacks. In the proposed hardware architecture the algorithm used for implementing the point multiplication unit is the Lopez -Dahab algorithm in [15]. The two major advantages of the Lopez-Dahab algorithm are (i) use of projective

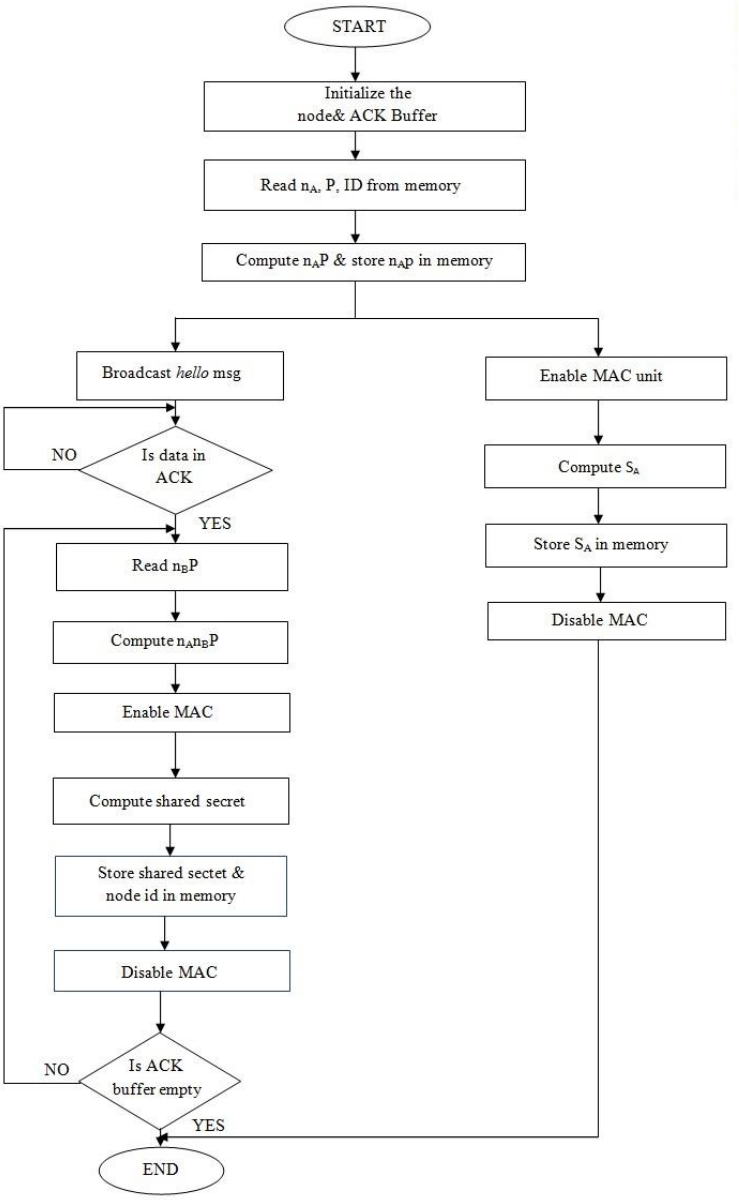

Figure 2. Flow chart of the ECKM system

co-ordinate representation helps in avoiding the complex finite field inversions and have reduced complexity since the variable y is not used (Field inversions and y value are required only for co-ordinate conversions) (ii) the number of finite field operations done in each iteration is independent of the key bit. This results in resistance against timing attack and Simple Power Analysis (SPA) attack.

The data flow graph of EC point multiplication algorithm used for implementation is shown in Fig 4. Data flow graph (DFG) is used to represent the sequencing flow of elliptic curve point multiplication algorithm's behavior graphically. DFG's are exploited to minimize the power dissipation, increasing the throughput, parallelization in operations etc. Finite field operations are represented using circles and each computation step is shown as a dashed line. The inputs participating in the finite field operation are shown at the top of each circle.

The result is shown at the bottom of each task and the outputs are shown in rectangles.

The hardware architecture of the point multiplication unit is given in Fig 5. The inputs to the ECPM unit are given through multiplexers with the select signals generated by the system controller.

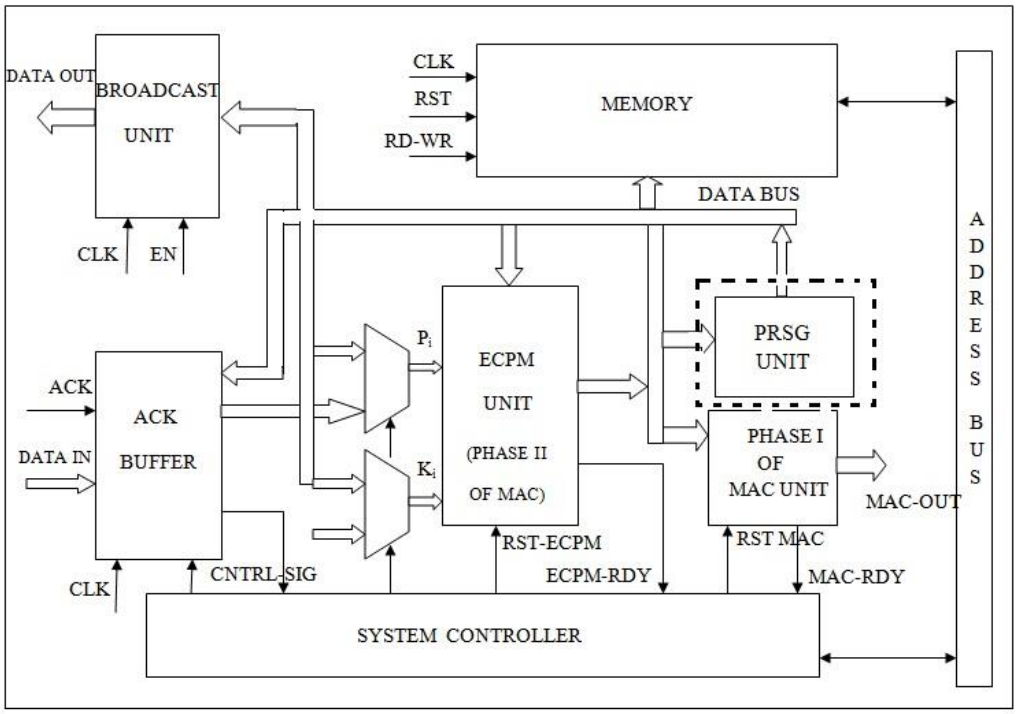

Figure 3. Hardware architecture

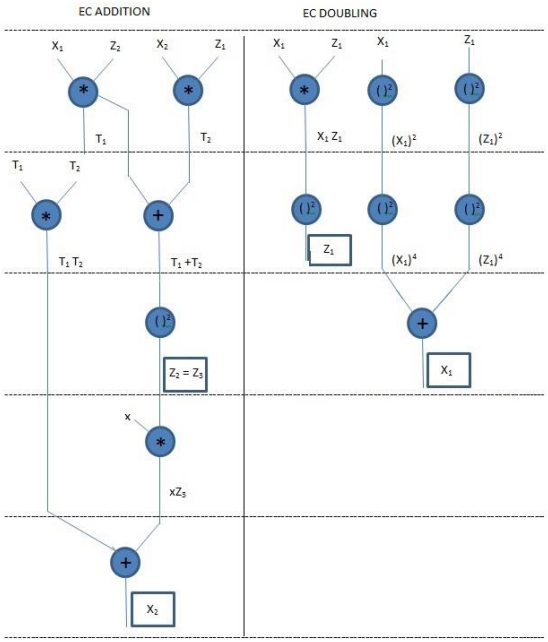

Figure 4. Data flow graph of EC point multiplication

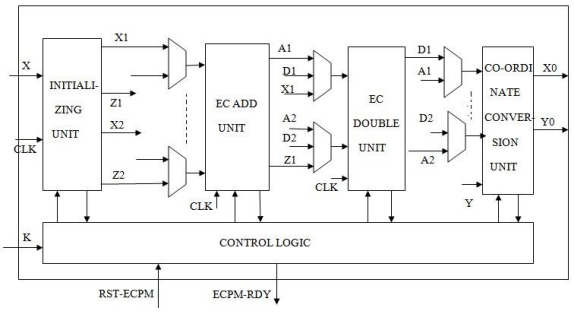

Figure 5. ECPM Unit

As shown in the architecture, the ECPM unit consists of an initializing unit, EC addition unit, EC doubling unit, co-ordinate conversion unit and a control logic. The finite field elements are represented using Gaussian normal basis to reduce the complexity of finite field operations. In the Gaussian normal basis representation field addition is a simple bitwise xor operation and squaring is a cyclic shifting which can be easily implemented using shift registers. The algorithm used for multiplication is the one cited in [16]. The field addition and squaring require a single clock cycle whereas finite field multiplication requires 163 clock cycles for implementation done over GF(2^{163}). The EC addition unit consists of two finite field

multipliers to reduce the time complexity of addition operation and the doubling unit consists of a single multiplier unit. The initialization unit is used to convert the affine x co-ordinate value into projective co-ordinate values x_1, x_2, z_1 and z_2. The inputs to the addition and doubling unit are given through multiplexers. Dependent on the key bits at each iteration, the control logic generates the select signals of the multiplexers. The EC addition and EC doubling are done simultaneously in contradiction to the double and add algorithm for point multiplication where these operations are sequential. The final output values x_1,x_2,z_1,z_2 and y co-ordinate are given to the co-ordinate conversion unit to generate the affine co-ordinate representation of the elliptic curve point. This conversion is required because, the commutative property of point multiplication is valid only in affine co-ordinate system. The coordinate conversion unit consists of a field inversion module which is the most complex finite field operation. The algorithm in [17] is used for implementing the field inversion unit and each field inversion operation involves nine finite field multiplication. This sequence of operations in the ECPM unit is controlled by the control logic which is designed as finite state machine to generate the control signals at each state.

5.2. Phase I of MAC unit

This unit consists of a Parallel-In-Serial-Out (PISO) shift register so that the input bits can be given serially to the two modular division circuits. When this unit is enabled, the generator polynomials for modular division are read from memory. The output of two modular division circuits are given to the GF unit to compute r' = k(r_1 + r_2). This r' is now fed back as the integer for point multiplication through a multiplexer.

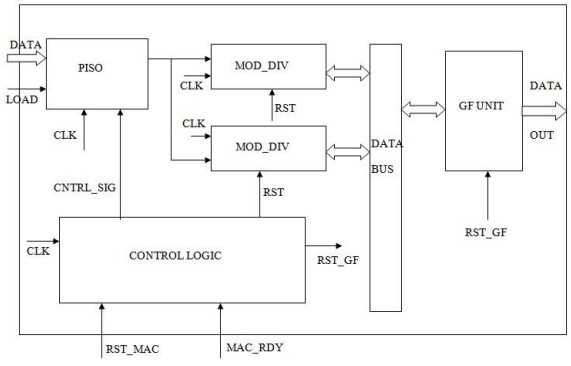

Figure 6. Phase I of MAC unit

5.3. PRSG unit

This unit is a part of the hardware for implementing modified EC-A protocol. The PRSG unit consists of an LFSR and an EC addition unit. The iteration key is generated as the xor of content of LFSR and the x-co-ordinate of the ECPM output. The EC point e_1P and the ECPM output are given to the EC addition unit to generate output sequence. The LFSR is clocked once in each iteration using a clock enable signal.

5.3. ACK buffer

The ACK buffer is used to store the ACK signals from the neighbouring nodes as a response to the hello message. The buffer unit consists of an input controller, an output controller and a FIFO structure to store the data. When an ACK signal is received, the write signal is enabled and the received data is written into the FIFO. The output controller checks for data in the buffer using the FIFO empty signal. When the data is in, the output controller enables the read signal and outputs the data. The output controller in the buffer unit is controlled by the system controller through buffer control signals. The size of ACK buffer is determined by the number of neighbouring nodes in the network. The system can also be modified such that when the buffer is full, the node transmits a wait signal to the neighbouring nodes to reschedule the transmission. Once the key establishment process is completed, this buffer can be used to store incoming data packets for further processing.

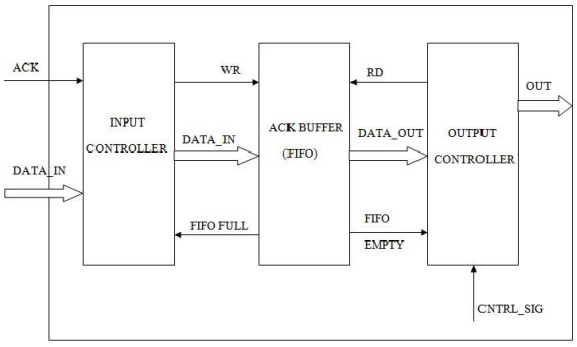

Figure 7. Acknowledgement buffer

5.4. Broadcast unit

The data to be transmitted is given out through the broadcast unit. This unit enables the RF section of the sensor node and transmits the data.

5.5. Memory

The secret keys and the data required for key establishment and authentication are stored in memory prior to deployment. At the end of the initialization phase the master key is erased and the shared secret keys and corresponding node ids are

stored in memory. The size of the memory required is determined by the number of neighbouring nodes in the network.

5.6. System controller

The system controller is designed as a finite state machine with a power on reset and generates the control signals for all the other units in the system.

5.7. Implementation results

The major units involved in the implementation of the proposed architectures are the ECPM unit, hardware unit for the first phase of MAC, ACK buffer and the system controller. In addition to the point multiplication unit, the hardware units required for implementing the MAC function are two modular division circuits and one finite field multiplier. The hardware for modified EC-A requires a PRSG unit which consists of EC addition unit and LFSR, in addition to the other units.

The proposed integrated architectures for ECKM and modified EC-A are implemented on a Kintex 7 board. In [3] the authors have suggested the use of AES (Advanced Encryption Standard) for constructing keyed hash (MAC) and SHA-1 for the evaluation of hash values, random number generation and as the key derivation function. For comparison, the modified EC-A using SHA and AES and ECKM using SHA for authentication are also implemented.

The resource utilization of various architectures when implemented on a Kintex 7 board is given in Table I. From the table it can be clearly seen that the resource requirement for implementation of ECKM is much less compared to the modified EC-A scheme. The implementation results also show that the structural complexity of any EC based key establishment scheme can be reduced by developing an integrated hardware rather than implementing standalone algorithms for each functionality. For example, implementation of the proposed integrated architecture for ECKM requires 11035 slice registers and 9758 slice LUTs whereas implementing SHA algorithm with ECKM requires 12105 slice registers and 10961 slice LUTs. Thus the integrated system helps to reduce 1070 slice registers and 1203 slice LUTs by sharing resources among operations of authentication and key exchange. Thus the proposed system is more suitable for WSNs in comparison with the implementation of standard algorithms.

Table 1. Hardware Requirement of Various Architectures

Logic Utilization	ECKM with EC based MAC	ECKM with SHA-512	Modified EC-A with EC based MAC & PRSG	Modified EC-A with SHA and AES
Number of slice registers	11035	12105	17576	18012
Number of slice LUTs	9758	10961	14170	25175

6. Conclusion

The hardware architecture and implementation results of two EC based key establishment techniques for WSNs are presented in this paper. Instead of standard algorithms, EC based algorithms are used for implementing MAC and PRSG and the EC point multiplication unit is time shared between these functions. The implementation results shows that use of EC based algorithms for authentication and random number generation reduces the structural complexity of the system to a large extent. The developed hardware can be interfaced with the sensor node through any serial communication port available in the sensor node.

7. Acknowledgements

This work was supported by the E-Security division of the Department of Electronics and Information Technology under Ministry of Communication and Information Technology of Government of India, as per order No.12(16)/2012-ESD dated 01-08-2013.

8. References

[1] M. Erol-Kantarci and H. T. Mouftah, “Wireless sensor networks for cost-efficient residential energy management in the smart grid,” IEEE Trans. Smart Grid, vol. 2, no. 2, pp. 314–325, Jun. 2011.

[2] S. Fernandez, P. De Toledo, and F. Del Pozo, "Usability and interoperability in wireless sensor networks for patient telemonitoring in chronic disease management." IEEE Trans. Biomed. Eng., vol. 60, no. 12, pp. 3331–3339, Sep. 2013.

[3] P. Kotzanikolaou, E. Magkos, D. Vergados, and M. Stefanidakis, "Secure and practical key establishment for distributed sensor networks," Security and Communication Networks, vol. 2, no. 6, pp. 595–610, 2009.

[4] J. Nam, M. Kim, J. Paik, Y. Lee, and D. Won, "A provably-secure ecc-based authentication scheme for wireless sensor networks," Sensors, vol. 14, no. 11, pp. 21 023–21 044, 2014.

[5] A. Moh'd, N. Aslam, W. Phillips, W. Robertson, and H. Marzi, "Snsec: a secure wireless sensor platform with hardware cryptographic primitives," Personal and ubiquitous computing, vol. 17, no. 5, pp. 1051– 1059, 2013.

[6] A. Mohammad, A. Gutub et al., "Efficient FPGA implementation of a programmable architecture for GF(p) elliptic curve crypto computations," Journal of Signal Processing Systems, vol. 59, no. 3, pp. 233–244, 2010.

[7] S. Liu, B. King, and W. Wang, "Hardware organization to achieve high speed elliptic curve cryptography for mobile devices," Mobile Networks and Applications, vol. 12, no. 4, pp. 271–279, 2007.

[8] P. Jilna and P. P. Deepthi, "Implementation of an elliptic curve based message authentication code for constrained environments," in Recent Trends in Computer Networks and Distributed Systems Security. Springer, 2014, pp. 520–529.

[9] P. Jilna and P. P. Deepthi, "Pseudorandom Bit Sequence Generator for Stream Cipher Based on Elliptic Curves," Mathematical Problems in Engineering, vol. 2015, Article ID 257904, 16 pages, 2015. doi:10.1155/2015/257904.

[10] S. Zhu , S. Setia , S. Jajodia, LEAP+: Efficient security mechanisms for large-scale distributed sensor networks. ACM Transactions on Sensor Networks, vol. 2, no. 4, pp. 500-528, 2006.

[11] L. Eschenauer, V.D. Gligor, " A key management scheme for distributed sensor networks", Proceedings of the 9th security, ACM Conference on Computer and Communications Washington DC, USA, 41–47 2002.

[12] D. Du, H. Xiong, and H. Wang, "An efficient key management scheme for wireless sensor networks," International Journal of Distributed Sensor Networks, vol. 2012, 2012.

[13] D. Singelee, S. Seys, L. Batina, I. Verbauwhede The Communication and Computation Cost of Wireless Security: Extended Abstract, WiSec '11, ACM (2011), pp. 1–4.

[14] P. Jilna and P. P. Deepthi, "A key management technique based on elliptic curves for static wireless sensor networks," Security and Communication Networks, vol. 8, no. 18, pp. 3726–3738, 2015.

[15] Y. Zhang, D. Chen, Y. Choi, L. Chen, and S.-B. Ko, "A high performance ECC hardware implementation with instruction-level parallelism over GF (2^{163})," Microprocessors and Microsystems, vol. 34, no. 6, pp. 228– 236, 2010.

[16] A. Reyhani-Masoleh and M. A. Hasan, "Efficient digit-serial normal basis multipliers over binary extension fields," ACM Transactions on Embedded Computing Systems (TECS), vol. 3, no. 3, pp. 575–592, 2004.

[17] R. Azarderakhsh, K. Jarvinen, and V. Dimitrov, "Fast inversion in with normal basis using hybrid-double multipliers," Computers, IEEE Transactions on, vol. 63, no. 4, pp. 1041–1047, 2014.

